2015 IEEE International Conference on Robotics and Automation (ICRA)
Washington State Convention Center
Seattle, Washington, May 26-30, 2015

A General Algorithm for Exploration with Gaussian Processes in Complex, Unknown Environments

Alberto Viseras Ruiz[1] and Calin Olariu[1]

***Abstract*— We propose a novel algorithm for efficient exploration with a single agent in unknown environments, populated with static obstacles. Every next position is computed by employing environment inference on top of path planning. The path planning process that ensures obstacle avoidance uses a modified version of the A* algorithm [1] and the inference is performed by direct sensing and by predicting values at yet-not-visited positions using Gaussian processes. We have validated our algorithm with densely measured data of an indoor magnetic field with significant spatial variations in different obstacle setups. Additionally, it is shown that it outperforms a previously developed algorithm [2], for obstacle-free scenarios, and the random movement of the agent.**

I. INTRODUCTION

A. Motivation

The need for humans risking their lives to explore unknown environments is diminishing as the deployment of intelligent agent networks is becoming more and more accessible. Specifically, exploring disaster management scenarios by a network of autonomous agents has been getting attention in recent literature. Our motivation stems from the fact that environments are unpredictable, which requires the combination of efficient exploration with obstacle avoidance, based solely on real-time measurements and on no prior knowledge.

B. Problem Statement

Existing approaches to exploration either treat the collecting of information in obstacle-free environments or make use of a prior belief about the environment. We wish to collect information efficiently and as quickly as possible, while navigating with a single agent in a completely unknown environment, populated with static obstacles. This requires the design of an algorithm to guide a deployed agent to minimize the overall uncertainty in a small amount of time.

C. Related Work

In [3], the authors treat the application of monitoring spatial-temporal dynamics with agents that have bounded resources. In order to maximize the information collected, they are proposing an approach (*eSIP* algorithm) to plan the informative paths, which is an *NP*-hard optimization problem. However, while their algorithm achieves near-optimal performance, it is non-adaptive i.e. they make use of the known belief about the environment which is, moreover, considered to be obstacle-free. Viseras *et al.* propose in [2] a multi-agent exploration algorithm that both is scalable and requires no prior knowledge about the environment. However, it assumes an obstacle-free environment. In [4], Julian *et al.* are focusing on a distributed robotic sensor network whose goal is to infer an environment. A gradient-based controller allows the multiple agents to move along the gradient of mutual information to minimize uncertainty and the authors proved it is locally optimal in most general form. The overall focus is on the distributive approach, as non-parametric methods are used for scaling w.r.t. the number of robots and a consensus is found to estimate joint measurement probabilities. However, the agents act in an obstacle-free environment. Our work employs a complex, populated, unknown environment and searches for paths to collect information efficiently. As new obstacles are detected during movement, the agent might need to change re-adapt the previously computed path.

There are many algorithms in literature capable of generating optimal paths, starting with Dijkstra's original algorithm [5] and incorporating further improvements. In [1], Hart *et al.* firstly describe the A* algorithm which uses heuristics to improve time performance. Moreover, Enhanced D* Lite outperforms A* in terms of computational complexity [6]. In addition, a series of algorithms (anytime planners) have emerged to deal with producing optimal plans while having time constraints [7]. Specifically, a highly suboptimal solution is found and then improved until a given time expires.

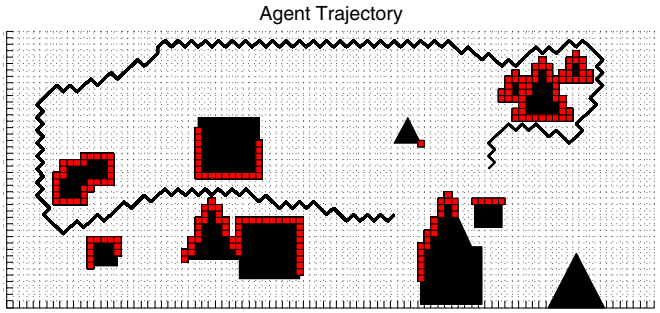

Fig. 1: Example of agent's trajectory in a 43×96 environment while exploring the magnetic field intensity in an indoor environment, running the novel algorithm. The static obstacles are represented by black blocks. The red cells represent the currently detected obstacles (current map of obstacles).

[1]Both authors are with the Institute of Communications and Navigation of the German Aerospace Center (DLR), Oberpfaffenhofen, 82234 Wessling, Germany, alberto.viserasruiz@dlr.de, calin.olariu1@gmail.com

In our case, we intend to adapt a low-complexity algorithm - A* - to suit the purpose of collecting information while avoiding collisions.

In [8], Zhu treats dynamic obstacle avoidance in an unknown environment. Specifically, he investigates motion models for non-static obstacles by predicting their movement using a Hidden Markov Model. In contrast, our work employs static obstacles while the information is predicted through Gaussian processes (GPs). A probabilistic approach on path planning with obstacles is tackled by Blackmore *et al.* in [9]. They reach the same complexity as in the case where uncertainty is not involved; however, information inference is not treated on top of the path planning process.

After this introduction, we derive a modified version of A*, used throughout the paper, and the background on Gaussian processes in Section II. The novel algorithm is discussed in detail in section III and the validation results are presented in Section IV. In the final section conclusions are drawn and future work is laid out.

II. PRELIMINARIES

A. Modified A* Algorithm

The A* algorithm [1] is widely used in path planning (also graph traversal applications) due to combining the core principle of both Dijkstra's algorithm [5] and Greedy BFS [10]: it is guaranteed to compute the shortest path and achieves good time performance. More exactly, it combines the knowledge g and the admissible heuristic h of every node x as the current node, into a cost function that drives the pathfinding:

$$f(x) = g(x) + h(x) \tag{1}$$

The Modified A* version we use is an adaptation from the classic A*. It does also take uncertainty of the positions s into consideration when expanding the path with the next node. Given x the current node, the successor[1] is found by:

$$x_{\text{succ}} = \underset{x_{\text{succ}}}{\operatorname{argmin}} \left(g(x) + \frac{d(x, x_{\text{succ}})}{\sigma^2_{x_{\text{succ}}}} + \frac{d(x_{\text{succ}}, x_{\text{target}})}{\frac{\sigma^2_{x_{\text{succ}}} + \sigma^2_{x_{\text{target}}}}{2}} \right) \tag{2}$$

with $d(x, x_{\text{succ}})$ being the straight distance from the current node to the candidate-successor and $\sigma^2_{x_{\text{succ}}}$ the variance of the candidate-successor. The second term is the heuristic to the successor and is expressed by the straight distance w.r.t the variance at the successor's position. The third term represents the heuristic to the target and is given by the distance successor-target w.r.t the average predicted variance between the extremities.

B. Spatial Gaussian Process Model

Consider, for example, the task of obtaining a map of the magnetic field intensity in an indoor environment. If we could model the spatial dependencies of the magnetic field well, we could fill spatial gaps between measurements with predictions. The stronger the dependencies are and the better they are represented in a model, the less measurements are needed to achieve a certain accuracy. Gaussian Processes have been used to model temperatures and other spatial phenomena [11]. It has also been shown, that the magnetic field intensity in an indoor environment can be modeled as a Gaussian Process [12].

A Gaussian Process is a collection of random variables, any finite number of which have a multivariate Gaussian distribution. It is fully specified by a mean function $m(\mathbf{x})$ and a covariance function $k(\mathbf{x}, \mathbf{x}')$, where $\mathbf{x}, \mathbf{x}'$ are vectors that define spatial positions [13].

We define the following vectors[2]: 1) $\mathbf{x} = [x_1, x_2, ..., x_n]$ is the n-observations input space. 2) $\mathbf{f} = [f_1, f_2, ..., f_n]$ is the set of the observed target values. Each of the observed target values corresponds to one of the variables in the n-observations input space. 3) $\mathbf{x}_*$ is the predictive input space.

Gaussian Processes are commonly used as priors in a Bayesian setting. Given $\mathbf{f}$ and $\mathbf{x}$, we can predict the target values $\mathbf{f}_*$ for the predictive input space $\mathbf{x}_*$. The elements in $\mathbf{f}_*$ are distributed according to: $p(\mathbf{f}_*|\mathbf{x}_*, \mathbf{x}, \mathbf{f}) = \mathcal{N}(\bar{\mathbf{f}}_*, \boldsymbol{\sigma}^2_{f_*})$. The mean vector $\bar{\mathbf{f}}_*$ and the variance vector $\boldsymbol{\sigma}^2_{f_*}$ of the posterior distribution are calculated as:

$$\begin{aligned} \bar{\mathbf{f}}_* &= m(\mathbf{x}_*) + K_*^T \cdot K^{-1} \cdot (\mathbf{f} - m(\mathbf{x})), \\ \boldsymbol{\sigma}^2_{f_*} &= K_{**} - K_*^T \cdot K^{-1} \cdot K_*. \end{aligned} \tag{3}$$

The matrices K, K_*, K_{**} are defined from the covariance function as: $K = [k(x_i, x_j)]_{i,j=1,...,n}$, $K_* = [k(\mathbf{x}_*, x_i)]_{i=1,...,n}$, $K_{**} = k(\mathbf{x}_*, \mathbf{x}_*)$.

The definition of the covariance function assumes the notion of similarity, which means that we expect that closer points are more likely to be similar. We focus our interest in stationary and isotropic covariance functions. The most widely used covariance functions in machine learning are the squared exponential (4) and the Matérn class (5) covariance functions. This second one is characterized by a smoothness parameter ν; with K_ν being the modified Bessel function and $\Gamma(\nu)$ the gamma function.

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \cdot exp\left[\frac{-(\mathbf{x} - \mathbf{x}')^2}{2l^2}\right] \tag{4}$$

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \cdot \frac{2^{1-\nu}}{\Gamma(\nu)} (\frac{\sqrt{2\nu(\mathbf{x} - \mathbf{x}')^2}}{l})^\nu K_\nu (\frac{\sqrt{2\nu(\mathbf{x} - \mathbf{x}')^2}}{l}) \tag{5}$$

The hyperparameters are a set of parameters which completely define the covariance function. The squared exponential and the Matérn class covariance function are defined by: 1) σ_f^2 as the maximum allowed covariance and 2) l modeling the covariance between variables separated by a certain distance.

Here, we define the mean function for the Gaussian Process $m(\mathbf{x})$ to be zero, since we do not want to assume a prior data model of the physical process.

[1] A successor is a neighboring, intermediate node between the current node and the target node.

[2] For simplicity in the notation, the formulation corresponds to a one-dimensional input space, which can be extrapolated to a multidimensional input space.

III. ALGORITHM DEVELOPMENT

Setup

We consider a scenario containing 3 elements: an environment, the information within the environment and a single agent.

We define the environment as the set of spatial positions $S = [s]$, $s \in \mathbb{R}^3$. The information contained in the environment is organized as information objects expressed by $O = [o]$, $o \in \mathbb{R}$ where each information object is placed in a spatial position $s \in S$. We consider a complex environment, i.e. one that is populated by obstacles that occupy the spatial positions $S_o = [s]$. The exploration is performed by a single agent which collects noiseless measurements in the set of measurements $MS = [o]$, with $o \in \mathbb{R}$. Additionally, it performs predictions and stores them as Gaussian distributions in the sets of means $PM = [pm]$, $pm \in \mathbb{R}$ and variances $PV = [pv]$, $pv \in \mathbb{R}_+$.

The algorithm consists of 5 tasks executed sequentially, at each step the agent takes in the environment:

A. Sensing

At every spatial position s reached, the agent takes one measurement (sensing) and stores it in MS:

$$\widehat{MS} = MS \cup \{o_s\},\ \forall s \in S_o \tag{6}$$

where o_s is the measurement at position s.
A spatial position may be sensed more than once, depending on the path the agent takes.

B. Prediction

The prediction process acts as an additional help at collecting information. Besides sensing, the uncertainty is also decreased by predicting values around the agent's path, based on measurements taken. We define the area around the agent's position where the means and variances are calculated as the prediction area:

$$PA_{R_p}(s_0) = \{s, \quad |s - s_0| \leq R_p\} \tag{7}$$

with s_0 the agent's position and R_p the prediction length.
The observations input space and predictive input space form the entire prediction area: $\mathbf{x} \cup \mathbf{x}_* = PA$.

The larger the prediction area, the more quickly the uncertainty decreases due to a larger observations input space. However, the complexity of computing the estimates grows cubically with the number of observations, which raises the issue of tuning PA depending on the available resources.

The resulting means and variances from PM and PV are updated at each step. During the movement, overlapping occurs (see Figure 2) between prediction areas at different steps:

$$PA_{\text{overlap}} = PA_{R_p}(s_0^{(l)}) - PA_{R_p}(s_0^{(k)}) \tag{8}$$

with $s_0^{(l)}$, $s_0^{(k)}$ the agent's position at steps l and k, respectively.
In situations like this, only the better predictions are updated:

$$pm_s = \begin{cases} o_s & \Leftrightarrow MS(s) = o_s \\ \overline{m}_{s_*} & \Leftrightarrow MS(s) \neq o_s \text{ and } \sigma^2_{s_*} < \sigma^2_{s_{old}} \end{cases} \tag{9}$$

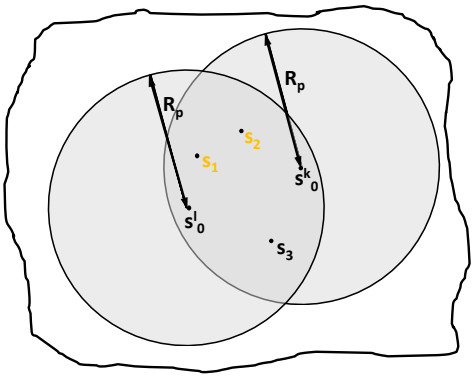

Fig. 2: Overlap of the prediction area at steps k and l; positions s_1, s_2 (orange) have better predictions at step l ($\sigma^{2\ (l)}_{s_1} < \sigma^{2\ (k)}_{s_1}$, $\sigma^{2\ (l)}_{s_2} < \sigma^{2\ (k)}_{s_2}$) and will be updated.

$$pv_s = \begin{cases} 0 & \Leftrightarrow MS(s) = o_s \\ \sigma^2_{s_*} & \Leftrightarrow MS(s) \neq o_s \text{ and } \sigma^2_{s_*} < \sigma^2_{s_{old}} \end{cases} \tag{10}$$

C. Scanning

The goal of the Scanning Process is the obstacle detection within the environment. This information is stored and used afterwards in the decision-making process in order to avoid obstacles. The scanning is defined by geometrical parameters that characterize the area swept by a laser scanner and the ray-emitting process (see Figure 3). In compact form, we have the 4-tuplet $(s_0, \theta, \alpha, r_s)$ denoting the agent's position (origin of rays), the angle view, the scanning resolution and the sensor range.

The process is repeated at every step in order to make decisions as early as possible.

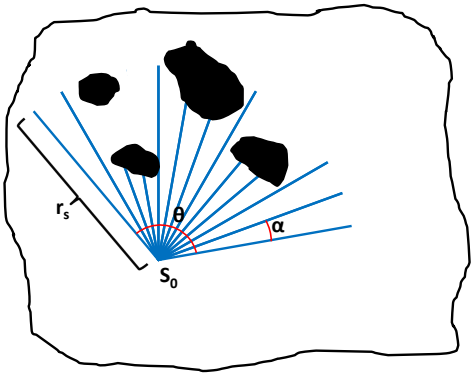

Fig. 3: The Scanning process. The agent's laser scanner emits rays that might intersect obstacles (black).

D. Decision Making. Next Position

In exploration we aim to decrease the uncertainty in the environment. Given every position s is characterized by a Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$, this translates into visiting positions with high variance.

P positions according to the highest P variances are chosen. We define a *P-combination* as an order of the P positions:

$$c_P = (p_{i_1}, p_{i_2}, \ldots, p_{i_P}) \tag{11}$$

with $i_1, i_2, \ldots, i_P \in \{1, 2, \ldots, P\}$.

Cost Function (C)

The combinations will be assessed by optimizing a cost function and then compared to find the order in which they will be visited. We design the cost function to embed the core behavior of the algorithm.

We pick the cost function as

$$C(p_{i_1}, p_{i_2}, \ldots, p_{i_P}) = \frac{H(p_{i_1}, p_{i_2}, \ldots, p_{i_P})}{t(p_{i_1}, p_{i_2}, \ldots, p_{i_P})} \tag{12}$$

where H denotes the information gained by the agent due to traversing combination $(p_{i_1}, p_{i_2}, \ldots, p_{i_P})$ and is calculated as the entropy of the combination. t is the total time to traverse combination $(p_{i_1}, p_{i_2}, \ldots, p_{i_P})$.

The path is computed by applying a Modified A* algorithm to the combination $(p_{i_1}, p_{i_2}, \ldots, p_{i_P})$. It determines a total covered area, by centering the prediction areas at all positions in the path. The entropy of the combination is calculated as the sum of the entropies of all positions within the area:

$$H(p_{i_1}, p_{i_2}, \ldots, p_{i_P}) = \sum_{i=1}^{i=|A_{comb}|} H(s_i) \tag{13}$$

with A_{comb} as the set of positions within the area covered by the combination $(p_{i_1}, p_{i_2}, \ldots, p_{i_P})$.

Velocity Profile

We define a *state* $\mathbf{x}$ as the tuplet $\mathbf{x} = \begin{pmatrix} s \\ v \end{pmatrix}$, with s the agent's position and v the agent's velocity. Based on the information collected through the scanning process, the algorithm calculates a velocity profile from the current state $\mathbf{x}_{\text{curr}}$ to the goal-states $\mathbf{x}_{\text{goal}}$ that are elements of the P-combination. In this manner, the agent can set its velocity at any moment in time so that the time t to traverse the combination is minimized. In this manner, the agent is enabled to accelerate when the current map of obstacles allows it to and decelerate in time to reach the required velocity at the next state.

Having the entropy H and time t of the combination, C can be calculated, as seen in (12).

Ordering

C is calculated for a predefined number of combinations (depending on the complexity we want to achieve) and the highest value indicates the order of the positions. The combinations that are compared are obtained by shuffling the positions inside.

E. Movement

Every combination has an associated path, calculated with Modified A*. Once the order is determined by the Decision Making process, the agent executes the first step on the path.

IV. EXPERIMENTAL VALIDATION

We test the proposed algorithm in a hybrid setup including real sensor data and subsequent simulation of the agent's movements and sensing. We use a personal computer running MATLAB[3] for our simulations. The information objects correspond to the magnetic field intensities in an indoor environment. We measure the magnetic field intensities using a robotic platform and a magnetic field sensor.

A. Experimental Setup

We use a holonomic robot equipped with an IMU to measure the magnetic field maps (see Figure 4). The robot is a modified version of the commercially available Slider platform by Commonplace Robotics. Due to its four mecanum wheels the platform is able to perform omnidirectional movements, following input commands for forward, lateral and rotational velocities.

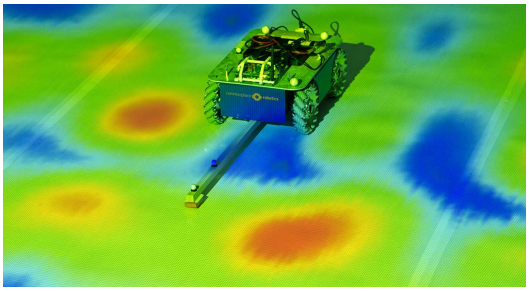

Fig. 4: Holonomic robot by Commonplace Robotics, equipped with a magnetic field sensor, generating a magnetic field intensity map. Here, we projected a visual representation of the measured magnetic field intensities on the lab floor.

The magnetic field sensor module used in the reported experiments is part of a commercial integrated sensor package (Xsens MTx). We mounted the sensor on a wooden beam that extended 0.75 m from the center of the robot. The purpose of this beam is to separate the sensor from the robot's ferromagnetic components and electromagnetic field generating devices.

We employ a commercial motion capture system (Vicon) to provide ground truth information of the robot's position. Our particular setup consists of 16 infrared sensitive cameras and infrared strobes.

B. Simulation Results

We execute the algorithm with a simulated agent. The information objects are based on a 40m^2 map of the magnetic field intensity of 4128 samples, captured on the ground within the DLR lab with a resolution[4] of 10 cm. The chosen resolution depends on the final application of the obtained map (in this case localization by using the magnetic field features [15]). We learn the hyperparameters of the Gaussian Process model by maximizing the marginal likelihood of a

[3]We use the GPML Toolbox [14] to carry out regression with Gaussian Processes.

[4]Spatial distance between two consecutive measurements.

subset of the information objects [13]. Then we use the remaining information objects for the experimental validation.

For simulations purposes, we considered a discretized, cell-based, 43×96 environment (grid), with the agent moving between the cells' centers. The prediction area is square-shaped with the side equal to $2R_p$. There are 4 types of obstacles that are randomly generated each run: triangle- and square-shaped in big and small sizes. The start position is also randomized each run to prove robustness of results.

1) Comparison with Random Movement in a random environment): It can be seen in Figure 5 that the novel algorithm outperforms a random movement of the agent as it reduces the mean square error (MSE) much faster. Given the exploration process is intended to take place in disaster management scenarios, having a quick reduction in uncertainty is of great importance.

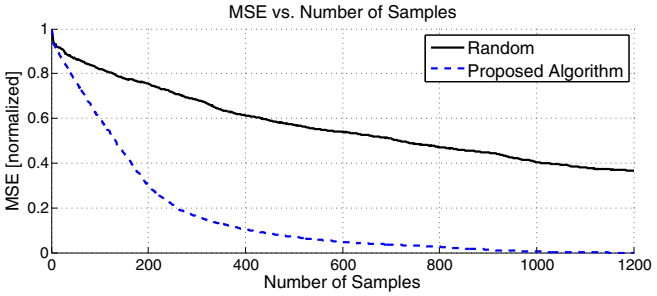

Fig. 5: Random movement vs. Proposed Algorithm. 50 runs, 15 obstacles (square side=triangle height=triangle base=5), $R_p = 11$, $P = 1$, $C = \frac{H}{t}$, Environment Size = 43×96.

2) Comparison with Random Movement in a particular environment (building floor): The 2^{nd} floor at the Institute of Communications and Navigation at DLR (see Figure 6) was used as a particular environment to exemplify the algorithm's performance (see Figure 7). The obstacles are the walls, stairs, etc. that compose that floor.

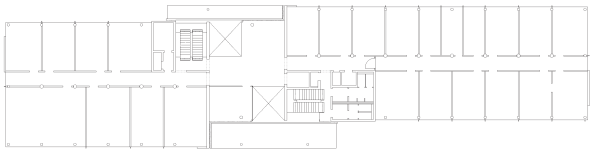

Fig. 6: Plan of 2^{nd} floor, Inst. for Comm. and Navigation, DLR.

3) Comparison with other algorithms in an obstacle-free environment: We also covered the obstacle-free scenario and in Figure 8 we compared the algorithm with a previously developed algorithm at DLR that performs exploration without taking obstacles into consideration [2].

4) Evaluation of the prediction length: The MSE reduces when increasing the prediction area, at the cost of increased complexity due to performing prediction in more positions. We illustrate the behavior of the algorithm in a random environment for multiple values of R_p in Figure 9. Depending on the size of the environment (here 43×96), there is a saturation point after which a larger R_p does not boost

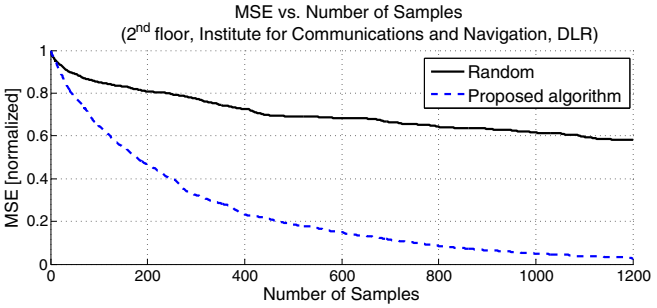

Fig. 7: Random movement vs. Proposed Algorithm. 50 runs, 15 obstacles (square side=triangle height=triangle base=5), $R_p = 11$, $P = 1$, $C = \frac{H}{t}$, Environment Size = 43×96.

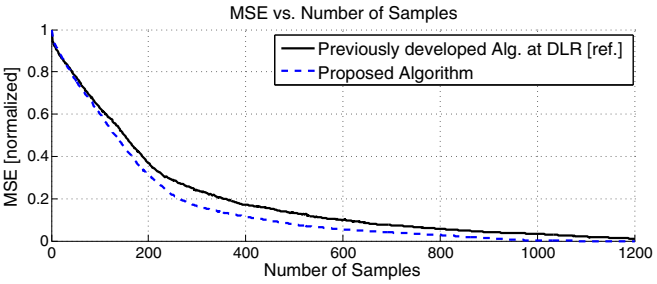

Fig. 8: Previously developed Algorithm [2] vs. Proposed Algorithm. 50 runs, no obstacles, $R_p = 11$, $P = 1$, $C = \frac{H}{t}$, Environment Size = 43×96.

performance anymore (see $R_p = 100$ (red) w.r.t. $R_p = 20$ (green)).

5) Evaluation of parameter P: In Figure 10 we investigate the decision making process by testing multiple values for the number P of highest variances that are chosen to be ordered and then visited and check which produce the best performance. Although no major difference is observed, choosing just the single highest variance tends to reduce the MSE faster; also, no comparison is needed, which yields a decrease in complexity.

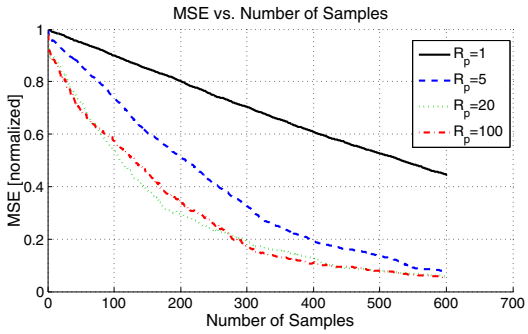

Fig. 9: Proposed Algorithm with different prediction lengths. 50 runs, 15 obstacles (square side=triangle height=triangle base=5), $R_p = 1, 5, 20, 100$, $P = 1$, $C = \frac{H}{t}$, Environment Size = 43×96.

6) Performance evaluation in increasingly occupied environments: Additionally, we are interested to validate the performance of the algorithm in scenarios where different percentages of the environment is obstructed by obstacles. Simulations are run 20 times for each percentage tested (see

Figure 11). It can be observed that no major differences occur between curves, which is proof for the stability of the performance. Independently of the amount of available information, the MSE decreases in time equally.

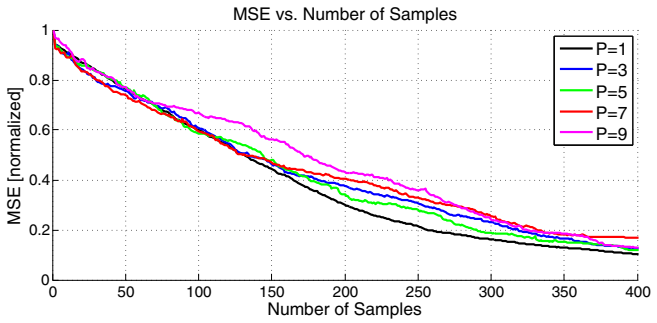

Fig. 10: Proposed Algorithm with different prediction lengths. 50 runs, 15 obstacles (square side=triangle height=triangle base=5), $R_p = 1, 5, 20, 100$, $P = 1, 3, 5, 7, 9$, $C = \frac{H}{t}$, Environment Size = 43×96.

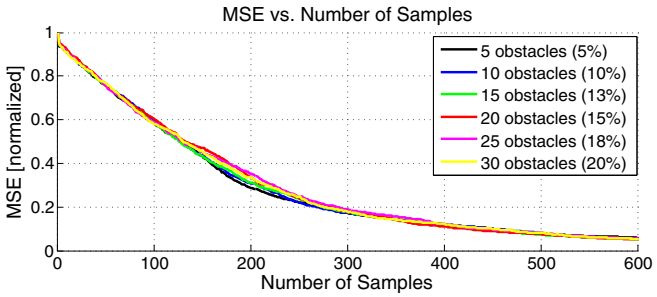

Fig. 11: Proposed Algorithm employed in an environment occupied in different percentages. 20 runs, Percentage occupied[%]: 5,10,13,15,18,20, $R_p = 11$, $p = 1$, $C = H, t$, Environment Size = 43×96.

V. CONCLUSIONS AND FUTURE WORK

We proposed and assessed a novel algorithm for efficient, single agent-based exploration in a complex, unknown environment. The inference of the environment is performed by both direct sensing and predicting through Gaussian processes, while the obstacle avoidance is covered by the Modified A* algorithm. We showed the algorithm outperforms the random movement of the agent and also a previously developed algorithm, used in obstacle-free environments. We confirmed the intuitive behavior that by increasing the prediction area the mean square error decreases faster in time and we came to the conclusion that we can achieve a relatively better performance (and less computation) by always choosing only one target with high variance. The algorithm is also stable in terms of performance when the agent is deployed in environments with different occupancy percentages.

Further work that stems from the algorithm and its setup includes employing multiple agents in the complex environment in order to share the information inferred. Therefore, we could use scheduling techniques in order to achieve higher performance by defining a global cost function common to all the agents. In order to do that, consensus algorithms should be investigated. In this work, we have not considered the uncertainties present in realistic scenarios. In future work we would like to take into account three types of uncertainties: pose, dynamics, and sensor uncertainties both to recognize the obstacles and to measure the physical process. In addition, we aim to derive algorithms that take into account the robot's dynamics constraints. One aspect that is subject to further investigations is the algorithm's computational complexity. More efficient path planning methods could be used to address this problem. One alternative which is scalable and efficient for high dimensional spaces are rapidly exploring random trees. In addition, in order to verify the applicability of the algorithm, we aim to test it with a robot in-the-loop.

ACKNOWLEDGEMENT

We would like to express our gratitude to Dmytro Bobkov, Christoph Manss and Joachim Mueller for their insightful suggestions.

REFERENCES

[1] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *Systems Science and Cybernetics, IEEE Transactions on*, vol. 4, no. 2, pp. 100–107, 1968.

[2] A. V. Ruiz, M. Angermann, I. Wieser, M. Frassl, and J. Mueller, "Efficient multi-agent exploration with gaussian processes," in *Australasian Conference on Robotics and Automation (ACRA)*, 2014.

[3] A. Singh, A. Krause, C. Guestrin, and W. J. Kaiser, "Efficient informative sensing using multiple robots," *Journal of Artificial Intelligence Research*, vol. 34, no. 2, p. 707, 2009.

[4] B. J. Julian, M. Angermann, M. Schwager, and D. Rus, "Distributed robotic sensor networks: An information-theoretic approach," *The International Journal of Robotics Research*, vol. 31, no. 10, pp. 1134–1154, 2012.

[5] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[6] S. C. Yun, V. Ganapathy, and T. W. Chien, "Enhanced d lite algorithm for mobile robot navigation," in *Industrial Electronics & Applications (ISIEA), 2010 IEEE Symposium on*. IEEE, 2010, pp. 545–550.

[7] M. Likhachev, D. Ferguson, G. Gordon, A. Stentz, and S. Thrun, "Anytime search in dynamic graphs," *Artificial Intelligence*, vol. 172, no. 14, pp. 1613–1643, 2008.

[8] Q. Zhu, "Hidden markov model for dynamic obstacle avoidance of mobile robot navigation," *Robotics and Automation, IEEE Transactions on*, vol. 7, no. 3, pp. 390–397, 1991.

[9] L. Blackmore, H. Li, and B. Williams, "A probabilistic approach to optimal robust path planning with obstacles," in *American Control Conference, 2006*. IEEE, 2006, pp. 7–pp.

[10] J. Pearl, *Heuristics*. Addison-Wesley Publishing Company Reading, Massachusetts, 1984.

[11] N. Cressie, "Statistics for spatial data," *Terra Nova*, vol. 4, no. 5, pp. 613–617, 1992.

[12] A. Kemppainen, J. Haverinen, I. Vallivaara, and J. Roning, "Near-optimal slam exploration in gaussian processes," in *Multisensor Fusion and Integration for Intelligent Systems (MFI), 2010 IEEE Conference on*. IEEE, 2010, pp. 7–13.

[13] C. E. Rasmussen and C. K. Williams, "Gaussian processes for machine learning (adaptive computation and machine learning)," 2005.

[14] C. E. Rasmussen and H. Nickisch, "Gaussian processes for machine learning (gpml) toolbox," *The Journal of Machine Learning Research*, vol. 11, pp. 3011–3015, 2010.

[15] M. Angermann, M. Frassl, M. Doniec, B. J. Julian, and P. Robertson, "Characterization of the indoor magnetic field for applications in localization and mapping," in *Proceedings of the International Conference on Indoor Positioning and Indoor Navigation (IPIN 2012)*, Sydney, Australia, Nov. 2012.